Name: Limpin,Jholim Jats R.
Section: NW-301

1. **Helloworld.sol**

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

contract FunctionIntro {
    // Functions can return multiple value.
    function add (uint x, uint y) external pure returns (uint) {    infinite gas
        return (x + y);
    }

    function sub(uint x, uint y) external pure returns (uint) {    infinite gas

    }
}
```

2. Value Types

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.7;

// Data types - value and references

contract ValueTypes {
    bool public b = true;
    uint public u = 123;

    int public i = -123;

    int public minInt = type(int).max;
    int public maxInt = type(int).max;
    address public addr = 0xf6ac838863A3031Bc9c0a1AB751199995f0F2d73;
}
```

3. Funtions

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

contract FunctionIntro {
    // Functions can return multiple value.
    function add (uint x, uint y) external pure returns (uint) {    infinite gas
        return (x + y);
    }

    function sub(uint x, uint y) external pure returns (uint) {    infinite gas

    }
}
```

Reflection:

In learning Solidity, I explored three important foundations: functions, HelloWorld, and value types. Functions are the actions of a contract, defining how data is read, updated, or manipulated, and they give life to otherwise static variables. The HelloWorld contract symbolizes the beginning of blockchain development, showing how a simple string and a function to return it can bridge curiosity with creation. Value types, such as booleans, integers, and addresses, represent the raw materials of smart contracts, capturing truth, quantity, polarity, limits, and identity. Together, these elements taught me that Solidity development starts with simple building blocks functions as verbs, HelloWorld as the first expression, and value types as the nouns and from these basics, complex decentralized applications are constructed.